

TOPDRIM deliverable 3.2

September 2014

Topological characterization of $S[B]$ systems: From data to models of complexityEmanuela Merelli^{*1}, Matteo Rucco^{†1}, Peter Sloom^{‡2}, and Luca Tesei^{§1}¹School of Science and Technology, University of Camerino, Italy²University of Amsterdam, The Netherlands. Complexity Institute, NTU, Singapore. ITMO University, St. Petersburg, Russian Federation**Abstract**

In this paper we propose a methodology for deriving a model of a complex system by exploiting the information extracted from Topological Data Analysis. Central to our approach is the $S[B]$ paradigm in which a complex system is represented by a two-level model. One level, the structural S one, is derived using the newly introduced *quantitative* concept of Persistent Entropy. The other level, the behavioral B one, is characterized by a network of interacting computational agents described by a Higher Dimensional Automaton. The methodology yields also a representation of the evolution of the derived two-level model as a Persistent Entropy Automaton. The presented methodology is applied to a real case study, the Idiotypic Network of the mammal immune system.

Keywords: Topological Data Analysis, Persistent Entropy, Transition Systems, Higher Dimensional Automata, Immune System, Complex Systems, Computational Agents.

1 Introduction

Complex systems are typically characterized by a finite, typically large, number of non-identical interacting entities that often are complex systems themselves, with strategies

^{*}emanuela.merelli@unicam.it

[†]matteo.rucco@unicam.it

[‡]p.m.a.sloom@uva.nl

[§]luca.tesei@unicam.it

and autonomous behaviors. Complex system science is a relatively young research area and it is raising interest among many researchers mainly thanks to the emerging of new techniques in several fields such as physics, mathematics, data analysis and computer science [51, 31].

Complex systems are analyzed using mainly two different approaches: the one that provides a global and abstract description by systems of differential equations [8], the other that focuses on the interacting components of a complex systems, which are generally simulated by an *agent based model and simulation framework* [26].

In terms of data volume a complex system can be associated to a big dataset and in order to extract from it useful information new techniques have been recently introduced, most of them in the area of *Topological Data Analysis* (TDA). TDA is largely used for the analysis of complex systems. Ibekwe et al. in [28] used TDA for reconstructing the relationship structure of E. coli O157 and the authors also proved that the *non-O157* is in 32 soils (16 organic and 16 conventionally managed soils). TDA was also used by De Silva in [15] for the analysis of sensor networks and it was successfully applied to the study of viral evolution in biological complex systems [12]. In [46] Petri et al. used a homological approach for studying the characteristics of functional brain networks at the mesoscopic level.

TDA is a new discipline inspired by homology theory. Informally, homology is a machinery for counting the number of *n-dimensional holes* in a topological space. A topological space can be formed by a collection of topological objects: the simplices (vertex, line segments, triangles, tetrahedra, and so on). A collection of simplices forms a simplicial complex [43]. Simplicial complexes can be built in several ways, e.g., by using the Vietoris-Rips complexes, Witness complexes, neighborhood complexes, clique complexes [10, 47, 7]. Simplicial complexes can be studied via *persistent homology* [17, 18] which computes a parametrized version of **Betti numbers**, represented by *Betti barcodes*. A barcode is a collection of line segments (or points in case of persistent diagrams [11]) where each line is equipped with two pieces of information: the spanning life time *intervals* of the homology class associated to the line and the *generators* of the homology class. An interval, written as $[i; j]$, means that the associated simplex forms a homology class at time $t = i$ and it survives until time $t = j$. If $j = \infty$, then the homology class is *persistent* and the interval is written $[i; +\infty)$; otherwise it is classified as *topological noise*. The generators are the set of simplices involved in the homology class.

In this work we propose a methodology for modeling complex systems that is based on TDA. Our contribution has been inspired by the ***S[B]*** paradigm [40, 39, 38]. Even if ***S[B]*** has been thought as a general theoretical framework for modeling complex *software* systems, we argue that it can be used to model all kinds of complex systems (biological, physical, chemical, and so on) with the following features:

- the system is formed by huge amount of interacting components, generally belonging to different classes and with different behaviors. In this kind of systems it is usually interesting to observe emerging global behaviors [26];

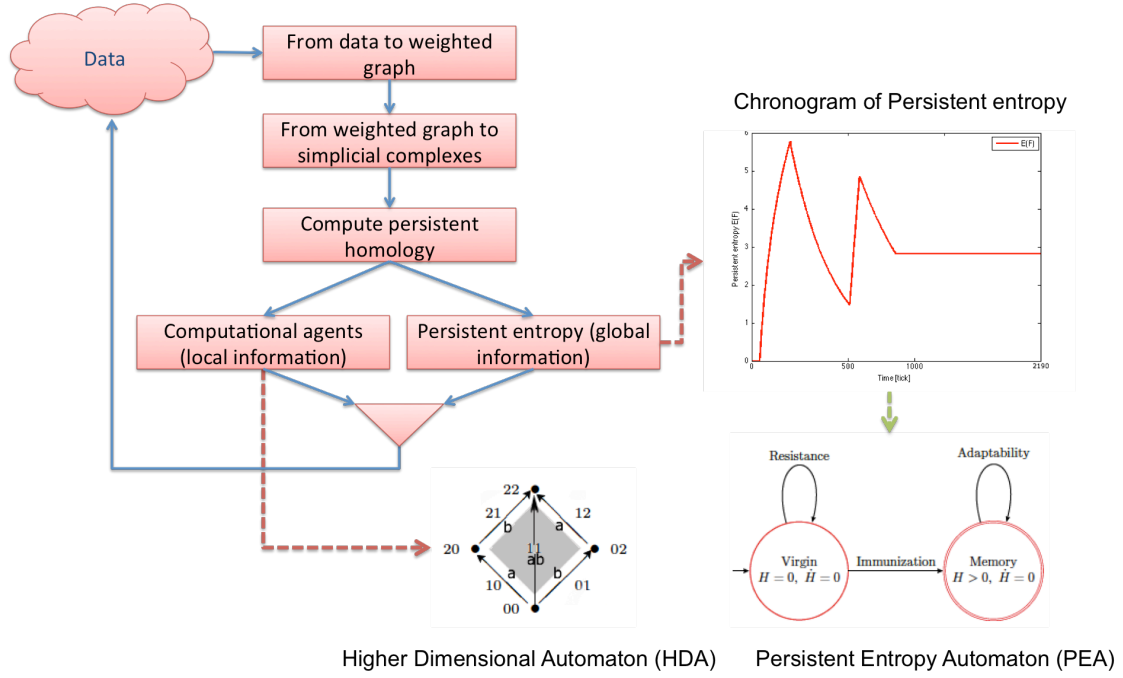


Figure 1: Graphical representation of our methodology. All the details are explained in the text, especially Persistent Entropy in Sect 3.4

- the system can be seen as the composition of two levels: a level characterized by the interacting components and a level containing the set of constraints and/or initial conditions that guide the system behaviour;
- the set of constraints can evolve over time. The time evolution of the set of constraints is well-known as *internal memory* or *adaptability* [19];
- all entities can simultaneously execute several actions in response to internal and/or external stimuli. The system behavior is like to that of a concurrent and distributed system studied in *concurrency* theory [42, 24, 2]
- each interaction can be characterized by a parameter that we call “proximity”.

Figure 1 graphically shows our methodology. It is an iterative process, where each iteration corresponds to a data observation taken from the complex system under study. In each iteration, data are first converted to a weighted graph in order to apply the subsequent steps. The graph is taken as the scaffold of a topological space from which a representation with simplicial complexes is derived. Then, persistent homology is computed in order to trace the topological invariants (Betti numbers) of the underlying topological space and the generators of the homological classes. At this point, two kinds of information are derived. On the one hand, the generators are taken as the more relevant computational agents among which local interactions in the complex system occur.

These agents are represented by a Higher Dimensional Automaton [49] and, within the $S[B]$ paradigm, they constitute an instantaneous description of the *behavioral level B*. On the other hand, the global dynamics of the complex system is described by a newly introduced entropic measure that we call *Persistent Entropy*. Its value for the current data is plotted on a chronogram. From the plot of persistent entropy a model of the global evolution of the complex system can be derived in the form of a *Persistent Entropy Automaton* that is a state machine whose transitions are constrained by the observed values of the persistent entropy and its first numerical derivative. Such an automaton, within the $S[B]$ paradigm, constitutes the *structural level S*.

To validate our methodology we consider, as case study, the *Idiotypic Network* (IN) of the *mammal immune system* [33]. This system has been largely studied because it exhibits all the features typical of a complex adaptive system (CAS) [26, 14] listed in the following.

- Distributed Control: Immune System (IS) is not centrally controlled. Detection and response can be executed locally and immediately without the need of communication of billions of immune molecules and cells with a central organ.
- Connectivity: IS is suited with inter-relationship, inter-action and inter-connectivity of the elements within a system and between a system and its environment.
- Co-evolution: the behavior of antibodies evolves depending on the behaviors of other antibodies; they can play a dual role.
- Sensitive Dependence on Initial Conditions: the IS reacts depending on the initial condition, e.g., the antigen volume
- Emergent-order: IS has both *self-regulation* and *self-protection* mechanisms.
- Learning and memory: IS is able to learn through its interaction with the environment.

Several models for the IN are available. In particular, we consider the simplified description modeled by Parisi [44] which is also the base for the agent-based simulator *C-ImmSim* [5]. More details about the model will be given in Section 2. In order to conduct a controlled experiment and facilitate the task of validation, in Section 4 we take as data observation of the complex system produced by the simulator, not real data. In this way we can check more easily the correctness of our methodology.

The paper is organized as follows. Section 2 introduces the case study of the IN. In Section 3 we describe our methodology step by step in order to extract from data a two-level model according to the $S[B]$ paradigm. In Section 4 we apply the methodology to the case study of Idiotypic Network and finally Section 5 provides concluding remarks and open issues to be faced in future works.

2 Case Study

In this paper we illustrate the steps of the proposed methodology applying it to a case study in biological Immune System (IS). Historically, the first theory is known as *Immune Network theory* (IN) by Nielse Jerne [33]. Jerne theorized that the immune system can be thought as a regulated network of antibodies and anti-antibodies, called “*idiotypic network*”. The network works even in the absence of antigens because the antibodies can be recognized as foreign cells, some of them previously stimulated by antigens. Let us briefly recall the main mechanisms described by the model. When an antigen is presented to the organism, the IS reacts following two ways: innate immunity and adaptive (or acquired) immunity. Innate immune defenses are non-specific, meaning these systems respond to pathogens in a generic way. The innate immune system is the dominant system of host defense in most organisms, it involves the eithelial barriers, the phagocytes, denditric cells, plasma proteins and NK cells. The adaptive immune system, on the other hand, is called into action against pathogens that are able to evade or overcome innate immune defenses. The first reaction of the immune system is governed by the mechanism of the innate immunity and its lifespan is up to 12 hours, after that the possible presence of antigens trigger the adpative response. Components of the adaptive immune system are mainly B-cells, antibodies, naive T cells and Effector T cells. When activated, these components “adapt” to the presence of infectious agents by activating, proliferating, and creating potent mechanisms for neutralizing or eliminating the antigens. The lifespan of the adaptive response depends by the type of infection. In this work we are interested to study the behavior of the antibodies involved during the adaptive response and we report a general example of this scenario. Suppose an antigen is recognized by B cells, which secrete antibodies, say Ab_1 . Ab_1 themselves are then recognized as anti-antibodies by “anti-idiotypic” B cells, which secrete other antibodies, say Ab_2 . Thus, further interactions can lead to Ab_3 antibodies that recognize Ab_2 and so on. In an idiotypic network, there is no intrinsic difference between an antigen and an antibody. Moreover, any node of the network can bind to and be bound to any other antigen or antibody. This phenomenon is known as *idiotypic cascade*. During the *onto-genesis* phase the IS learns which antibodies should not be produced and the system remembers these decisions for its entire life. This phenomenon is called *immunological memory*. Important properties of this system, including memory, are then properties of the network of cells as a whole, rather than of the individual cells [25]. This phenomenological description has been formalized by Parisi [44]. Parisi derived a simplified model for describing the dynamics of a functional network of antibodies in absence of any driving force of external antigens, namely when the concentrations of antibodies are time independent, e.g. during the immune memory state. The dynamics h_i of each antibody at the equilibrium is simply described in Equation 1:

$$h_i = S + \sum_{k=1,n} J_{i,k} c_k \quad (1)$$

Where $J_{i,k}$ ($J_{i,i} = 0$; $J_{k,i} = J_{i,k}$) represents the influence of antibody k on antibody i . If $J_{k,i}$ is positive, antibody k triggers the production of antibody i , whereas if $J_{k,i}$

is negative, antibody k suppresses the production of antibody i . $|J_{k,i}|$ is a measure of the efficiency of the control of antibody k on antibody i . The $J_{i,k}$ are distributed in the interval $[-1; +1]$. S is a threshold parameter, that regulates the dynamics when the couplings $J_{i,k}$ are all very small; otherwise one can assume S equal to zero. The concentration c_i of antibody i is assumed to have, in absence of external antigens, only two values, conventionally 0 or 1 (in the presence of antigen concentrations c_i might become ≥ 1). The immune system state is determined by the values of all c_i 's for all possible antibodies ($i = 1; \dots; N$). h_i represents the total stimulatory/inhibitory (depending on its sign) effect of the whole network on the i -th antibody. h_i is positive when the excitatory effect of the other antibodies is greater than the suppressive effect and then c_i is one. Otherwise h_i is negative and c_i is zero.

3 Methodology

The aim of our methodology is to extract local and global information from data applying TDA. It is suitable for studying the class of complex systems described in Section 1. In particular, the output of the methodology is a set of models set up within the $S[B]$ paradigm [40, 39, 38]. In the $S[B]$ paradigm a model is specified using two levels of description, namely the S *global* or *structural* level and the B *local* or *behavioural* level, which are entangled in order to express the behavior of the system as a whole. The S level describes how the system evolves following global information coming from the environment in which it is operating and from the interactions and the evolutions of the model entities of the B level. Note that this approach has some similarities with hierarchical models like *Hierarchical Automata* proposed by Mikk et al. in [41]. However, in our case the levels can not be seen as just different levels of abstraction, but they can exist only in the entangled $S[B]$ version.

3.1 From Data to Weighted Graphs

We start from data that comes from observations, over time, of the complex system under study. The first step to be accomplished is to represent such data as weighted graphs where:

- nodes represent the interacting components;
- an arch, equipped with a weight, expresses an interaction (or a distance) between two components.

Note that this is always possible; for instance the weight can be defined using statistical descriptors (correlation coefficients, scoring systems, and so on), metric functions (normalized Euclidean distance, Hamming distance, and so on) or domain dependent features.

Graphs are strictly related to topological spaces; in particular a graph can be thought as the skeleton of a topological space. Let us now introduce the description of a topological space that will be useful for our purposes.

Definition 3.1. (Topological Space)

A topological space \mathbb{X} can be described by a possibly infinite set of pairs of the form (β_n, V_n) , $n \geq 0$, where:

- β_n is the n -th Betti number
- $V_n \subseteq H_n$ is the set of generators of the homological class H_n , represented using an appropriate notation.

Concerning the notation for V_n , in case of V_0 we will use natural numbers, e.g. $\{0, 1, 2, 3\}$, to identify the 0-simplices and in case of V_1 we will use a geometrical representation of the simplicial complex through an edge list, e.g. $[0, 1] + [1, 2] + [2, 3] + [3, 0]$. In case of V_2 , a tetrahedron is the composition of triangles, e.g. $[0, 1, 2] + [0, 1, 3] + [1, 2, 3] + [0, 2, 3]$. In the rest of the paper, we use the notation $v \subseteq v'$ for indicating that v is a simplex of dimension strictly less than the dimension of the simplex v' and it is a part of v' . For instance, in the example above $2 \subseteq [1, 2]$, $[0, 1] \subseteq [0, 1, 3]$ and $0 \subseteq [0, 1, 3]$. For a more formal definition we refer to [20].

3.2 Form Weighted Graphs to Simplicial Complexes

Given a directed or undirected graph, it is possible to construct from it a simplicial complex following several approaches [34]. In this methodology we apply the *clique weight rank persistent homology* (CWRPH) [47]. This innovative technique is based on the concept of a flag complex: given a graph G , the simplices of a clique complex $C(G)$ are the complete subgraphs of G and the 0-simplices of $C(G)$ are the vertices of the subgraphs, (i.e., the complete subgraph complex). The maximal simplices are given by the collection of the vertices that make up the cliques of G . In the literature, a clique complex is also referred to as a flag complex. CWRPH describes a formal procedure for building a $C(G)$ from a weighted graph.

jHoles is the first Java high-performance implementation of the CWRPH algorithm [7]. It implements the following standard clique weight rank persistent homology:

- 1 extract the descending (or ascending) list W of all weights w_t indexed by the discrete filter parameter t ;
- 2 list all maximal cliques of each connected component in G ;
- 3 find all the sub-cliques of each maximal clique;
- 4 for each sub-clique and maximal clique, rank it according to the index t of the minimum (maximum) weight; note that these are clique simplicial complexes equipped with a filter value;

In Figure 2 it is shown an example of application of the CWRPH algorithm implemented in *jHoles*. Given the undirected weighted graph on the left, with weights W , *jHoles* finds four 3-maximal cliques shown on the right with different colors; each of them is equivalent to a 2-simplex (filled triangle).

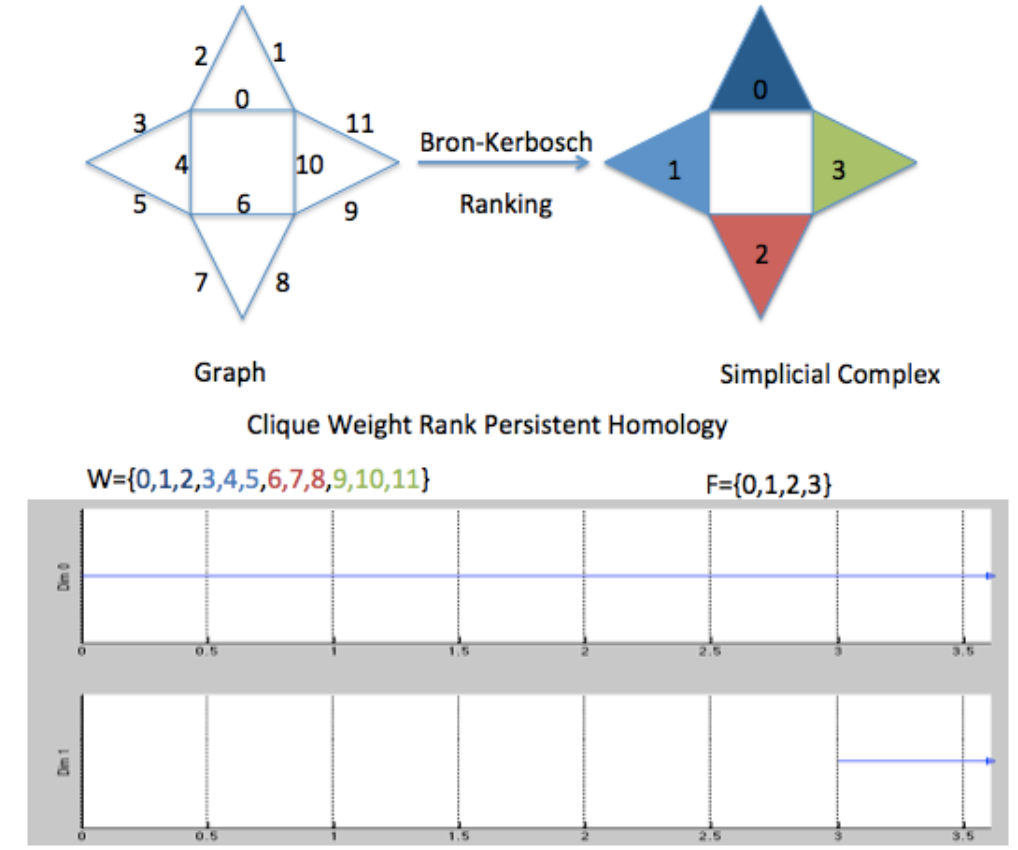


Figure 2: jHoles application example. Construction of a topologicFlal space from an undirected weighted graph (up). The Betti barcode representing the evolution of topological invariants (down). Where W and F denote weights and filter values.

3.3 From Simplicial Complexes to Computational Agents: Local Interactions

The last step of the jHoles computation takes the clique simplicial complexes calculated in step 4 (see Section 3.2) and computes the persistent homology giving as output Betti barcodes, intervals and generators.

Consider again Figure 2. The simplices are labeled with a filter value that is used as index during the computation of persistent homology [53]. Roughly speaking, the algorithm builds the topological space by including the simplices in the order given by their appearance filter value. In the example, the algorithm runs 4 iterations; at each step it introduces a simplex (giving filter values $F = \{0, 1, 2, 3\}$) that eventually is connected to the simplices previously introduced. Moreover, it computes the topological invariants (Betti numbers) of the newly constructed topological space. The pairs formed by Betti numbers and filter values are graphically represented with a Betti barcode. In this example the Betti numbers are $\beta_0 = 1$ meaning that there is only one connected component and $\beta_1 = 1$, indicating that a 1-dimensional persistent topological hole is present. This hole is formed by four 1-simplices and four 0-simplices, which are its generators. The connected component appears at filter value 0 in correspondence with the first 2-simplex indexed 0 and persists throughout the whole filtering, as indicated in the DIM 0 of barcode in Figure 2. The persistent hole appears after that the four 2-simplices are connected together at filter value 3 (DIM 1 of barcode). Note that in this example all the lines in the barcodes are persistent. However, this is not the case in general, because topological noise can be present (see Section 2).

At this point of our methodology, we use the generators of persistent homological classes of the derived topological space. In our point of view the 0-simplices in the generators represent the more relevant interacting components of the complex system under study. They interact with other (homogeneous or heterogenous) entities or with the environment in which the system is immersed. The n -simplices in the generators then tell how these components are connected together in the system. This is the information that can be automatically derived with the proposed methodology.

It is very natural, in this view, to use the concept of *Computational Agent* [32, 8] to represent the identified more relevant interacting components. We want to derive a computational model for studying their behavior. An agent can execute a generic number of actions, possibly concurrently. However, TDA extracts information about connectivity among agents, but it does not provide any information regarding the actions executed by the entities. Thus, for obtaining a complete operational model of the interactions there is still the need of using domain-based knowledge. For instance, in Section 4 we will use the Jerne model described in Section 2.

Among other computational models available for expressing concurrent computations, in this case the *true concurrent* characteristic of the kind of systems considered (not CPU-based), suggests the use of Higher Dimensional Automata for modeling the behavior of the agents.

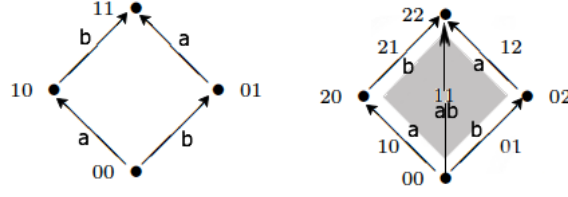


Figure 3: A classical Finite State Automaton representing the interleaving of actions a and b (left). The corresponding Higher Dimensional Automaton (right).

3.3.1 Higher Dimensional Automata

The problem of constructing a convenient model of computation that takes into account both the aspect of true concurrency, or non-interleaving concurrency, has been solved by using a geometrical approach. Several studies regarding a geometrical description of concurrency and automata have been published [37, 6, 35].

In 1991 Pratt published the formal definition of a Higher Dimensional Automaton based on the concept of n -complexes [49]. This model is a generalization of automata to allow them to express non-interleaving concurrency. Pratt generalized the standard Finite State Automaton definition, based on states and edges, to general n -dimensional objects where n is any finite dimension. The basic idea is that an n -dimensional object stands for a n -dimensional transition representing the concurrent execution of n actions.

A concept to manage with care is that of a “hole”. Suppose to have the automaton on the left part of Figure 3. The initial state 00 has two outgoing edges labeled a and b , respectively, followed by b and a . Under the interleaving semantics, the automaton accepts both the words ab and ba . However, such automaton contains a hole, indeed the interior of the square. Pratt suggested to replace the interleaving semantics of this automaton, towards true concurrency, by *filling the hole*. The new surface, depicted on the right part of Figure 3, represents the simultaneous execution of a and b without imposing any order. Conversely, the “sculpting” of the inner surface by proper transformations allows to represent with HDAs also classical *non-determinism*.

The initial formal definition of HDAs by Pratt can be found in [49]. However, in the subsequent papers [21, 48], a more intuitive and concise definition was given by using Chu spaces. For the sake of simplicity, we recall here this definition. A Chu space [48] is a structure over an alphabet Σ , namely a rectangular array whose entries are drawn from Σ . $\Sigma = \{0, 1\}$ is appropriate for representing ordinary event structures [54], whose events may be either *unstarted* or *finished*. The alphabet can be extended to $\Sigma = \{0, 1, 2\}$ meaning that an action can be *unstarted*, *executing* or *finished*. The 3-length alphabet is used for HDAs.

Definition 3.2. (Chu Space)

A Chu space \mathbb{A} is a tuple (A, r, X) over an alphabet Σ where:

- A is a set of *actions* or *events*;
- X is a set of *states*;
- $r: (A \times X) \rightarrow \Sigma$ is a function representing whether or not an event belongs to a state.

Consider the FSA on the left part of Figure 3. Its representation as a Chu space with $\Sigma = \{0, 1\}$ can be given with the following matrix representing the function r of the Chu space:

a	0	0	1	1
b	0	1	0	1

Every column represents a possible state of the automaton. For instance, state 00 is the initial one where both a and b are unstated, state 01 is the one in which a is still unstated and b is finished, and so on.

Consider now the HDA on the right part of Figure 3. Its representation as a Chu space with $\Sigma = \{0, 1, 2\}$ can be given as follows:

a	0	0	0	1	2	1	1	2	2
b	0	1	2	0	0	1	2	1	2

State 01 now is the one in which action a is unstated and action b is executing, state 12 is the one in which action a is executing and action b is finished, and so on.

Chu spaces have also been used for the study of concurrent programs. In [16] Du et al. proposed an enriched process algebra for the Chu spaces for studying the concurrency of object-oriented programming languages. In [30, 29] Ivanov presented an algorithm for generating Chu space models for describing the behaviors of complex non-iterated systems, including n -ary dependencies.

3.3.2 Derivation of HDAs from Topological Generators

The next step of the methodology is to derive from the simplicial complex constructed in Section 3.2 an HDA representing the behavioral level of the model within the $S[B]$ paradigm. Such HDA is the parallel composition of the HDAs representing the behaviour of each computational agent identified by the 0-simplices in the generators V_0 . For the HDA we will use the Chu space representation. It is useful for our purposes to define the set A of the Chu space representing the HDA in such a way that the name of the actions carry additional significant information.

Let \mathbb{X} be representation of a topological space as in Definition 3.1, i.e. pairs of Betti numbers β_n and the corresponding generators V_n of the homological classes.

Definition 3.3. (Labels of Actions)

The labels A of the Chu space representing the HDA are of the form (action, source, target) where:

- action is a description name;
- source and target are the identifiers of two 0-simplices $v, v' \in V_0$ and there exists $v_n \in V_n$ ($n > 0$) such that $v \in v_n$ and $v' \in v_n$.

Note that in general, when no particular constraints are present in the domain knowledge, the actions are bi-directional, i.e. there is an action a_0 from v to v' and the same action a_0 from v' to v . Moreover, the domain knowledge should suggest a certain number of actions and assign to them a meaningful name.

3.4 From Simplicial Complexes to Persistent Entropy: Global Information

Let us consider again the Betti barcodes, intervals and generators calculated from simplicial complexes as described at the beginning of Section 3.3. In order to extract global information from the data-derived topological space, the next step of the methodology prescribes to compute a newly introduced notion that we call *persistent entropy*. This entropy measure is basically calculated using the persistent Betti barcodes and the topological noise of the underlying topological space. By definition, the value of the persistent entropy is strongly related to the topological structures derived from data.

Diaz et al. defined an entropy based on the persistent barcode (Definition 3 of [13]). The aim of their paper is the definition of an entropy-driven algorithm for finding the best filtration of a set of simplices. We argue that when the filtration is given their entropy can be easily extended without losing the interpretation à la Shannon. Here we propose to use the maximum of the filtration value plus one as upper bound of a persistent barcode.

Definition 3.4. (Persistent Entropy) Given a filtered topological space, let F be the set of its filtration values and let $J = \{0, 1, \dots, n\}$ be a set of indices for the lines appearing in the whole barcode, independently from the dimensions. A line corresponding to topological noise is denoted by $[a_j ; b_j]$, with $j \in J$. Instead of $[a_j ; \infty)$, a persistent topological feature is denoted by $[a_j ; b_j = m)$ where we set $m = \max\{F\} + 1$.

The *Persistent Entropy* H of the topological space is calculated as follows:

$$H = - \sum_{j \in J} p_j \log(p_j)$$

where $p_j = \frac{l_j}{L}$, $l_j = b_j - a_j$, and $L = \sum_{j \in J} l_j$.

Note that the maximum persistent entropy corresponds to the situation in which all the lines in the barcode are of equal length. Conversely, the value of the persistent entropy decreases as more lines of different length are present.

Consider again the topological space derived in Figure 2. The set $F = 0, 1, 2, 3$, let $J = \{0, 1\}$ where 0 is the index of the line in DIM 0 and 1 is the line in DIM 1, both persistent in this case. Then, $m = 4$ and the lines are $[0, 4)$ and $[3, 4)$ yielding $H = 0.5$.

3.5 The Derived $S[B]$ Model

In the vision of the $S[B]$ paradigm global information constraints local interactions within a certain subspace of all the possible interactions according to equilibrium conditions (steady states) characterizing the system. Whenever external or internal factors alter this equilibrium, the system reacts by entering an “adaptation” phase towards a new equilibrium [38]. Thus, if we observe the system at any moment, like we do when we take samples of the data, it will be either in a stable condition or in an adapting phase because the global information (strictly bond to the current local configuration) is violating the current equilibrium condition. As the system evolves, it can go back to the previous steady state or it may eventually reach a new kind of known or unknown equilibrium.

In our methodology persistent entropy is the global information that, at the S level, naturally constraints the local interactions at the B level represented by the HDA constructed as described in Section 3.3. Note that we derive from data and from domain specific knowledge a set of models each of which give an $S[B]$ representation of the system at the moment of the observation. The extraction of the rules governing the evolution of the system between one observation and the subsequent one is beyond the scope of this work.

3.6 Global Evolution of the Extracted Models

The evolution over time of the persistent entropy can be used for detecting when the system reacts to stimuli, i.e., starts an adaptation phase. Let us consider the chronogram plotting the values of the persistent entropy along the observed sequence of data of the system. Indeed, a peak in the chart means an abrupt change in the topology of the simplicial complexes. This information may be used to find the intervals *before*, *during* and *after* a stimuli (or a phase transition) [22]. On the other hand, a plateau in the chart corresponds to an equilibrium condition, namely a steady state. The relevant regions can then be used for deriving a state machine representing the global observed evolution of the system.

Automata [27] are one of the mostly used formalism for modeling systems (hardware, software, physical systems, and so on). They are a very familiar and useful model that extends the concept of directed graph associating to the nodes and the edges the meaning of *states* and *transitions*, respectively. A state describes a static feature of the behavior of the system during its evolution, while a transition models how the system can evolve from one state to another. In the literature, several types of automata have been discussed [45, 1, 23, 50]. Among others, a generalization of the formalism of automata and, more generally, transition systems is that of *Uniform Labeled Transition Systems* (ULTraS) [3, 4]. In this paper our automata are equipped with topological information coming from data.

Following the approach of [38], we use a notion of *observable* that is derived from the behavioral level B and is used in S to define constraints associated to the S states. Such constraints identify the known *steady states* of the $S[B]$ system and, when violated, put

the whole system in an *adapting* phase.

Definition 3.5. (Persistent Entropy Automaton)

A Persistent Entropy Automaton (PEA) is a tuple $S = (R, \Lambda, r_0, H, \rightarrow_S, L)$ where:

- R is a set of steady states;
- Λ is a set of labels;
- $r_0 \in R$ is the initial steady state;
- H is the observable variable, corresponding to the current value of Persistent Entropy;
- $\rightarrow_S \subseteq R \times \Lambda \times R$ is a labelled transitions relation;
- $L: R \rightarrow \Phi_H$ is a labeling function associating to each state $r \in R$ an *invariant condition* on H representing the global constraints that characterize the equilibrium configuration expressed by r .

4 $S[B]$ Model of the Case Study

Accordingly to Jerne [33], the Idiotypic Network (IN) can be described by three main states: *virgin*, *activation* and *memory*. In the *virgin* state there are not antibodies but only non-specific T-cells that start their activities after the activation signal sent by B-cells. B-cells recognize that the environment has been disrupted by pathogens. Then, the IN *proliferates* antibodies and reach the *activation* state, which is not a steady state. After the activation, the IN performs the *immunization* during which the antibodies play a dual role. In fact, an antibody can be seen both as a *self* protein (a protein of the organism) or a *non-self* protein (a pathogen to be suppressed). After the *immunization* action the IN reaches the *memory* state. This state represents a steady condition in which there is only a selection of antibodies. All the *transitions and states* of IN are characterized by the fact that antibodies perform basically two actions: *elicitation* and *suppression*.

In the rest of this section we report about the application of our methodology to the IN simulated with *C-ImmSim* [5, 9, 36]. We executed several (in the order of hundreds) simulations, each of them characterized by:

- a lifespan of 2190 ticks, where a tick corresponds to three days;
- a repertoire of at most 10^{12} antibodies, i.e., the maximum number of antibodies available during the whole simulation;
- an antigen volume $V = 10\mu L$.

During the simulation an antigen is injected and, after an unknown (simulated) transient period of time, the same antigen is injected again.

4.1 From data to weighted graph

In *C-ImmSim* each idiotypic (both antigens and antibodies) is represented with a bit-string, in our case of 12 bit length. Two idiotypes A_j and A_k interact if and only if their Hamming distance $d(A_j, A_k)$ is such that $11 \leq d(A_j, A_k) \leq 12$. The pair-wise distances among all the idiotypes are stored in a matrix J , the so-called *Affinity matrix*.

However, the so formed affinity matrix is not very realistic because it does not take into account the volume of the idiotypes. For this reason, in this case study we decided to replaced it with a *Coexistence Matrix* C where each element is a coexistent index.

Definition 4.1. (Coexistence Index)

Given the Hamming distance $d(Ab_j(t), Ab_k(t))$ between two antibodies and their volumes $[Ab_j(t)], [Ab_k(t)]$ at tick t , their *coexistence index* is defined as follows:

$$C_{Ab_{j,k}}(t) = \frac{d(Ab_j(t), Ab_k(t)) \cdot [Ab_j(t)] \cdot [Ab_k(t)]}{\sum_{l=1}^n [Ab_l(t)]} \quad (2)$$

Equation 2 expresses the fact that for lower values of affinity the volume must be more significant because the match between antibodies is less probable.

The coexistence matrix C is a symmetric matrix and each element is taken to represent a weighted arc of an undirected graph. In this way we obtained a graph representation of our data taken from the simulation. Figure 4 shows the weighted idiotypic network at the end of the simulation.

4.2 From Weighted Graph to Simplicial Complex

We computed the coexistence matrix (see Equation 2) and we used the weighted idiotypic network as the input for the persistent homology computation. As indicated by the methodology we used the CWRPH, recent development in TDA, providing a new approach to the study of weighted networks. One of the main advantages of this approach is that it preserves the complete topological and weight information, allowing to focus on special mesoscopic structures, e.g., weighted network holes, which connect the network's weight-degree structure to its homological backbone [47].

We used jHoles [7] to process our weighted graph towards the production of the simplicial complex as described in Section 3.2, points 1-4 of the jHoles algorithm.

4.3 Computing Persistent Homology

The persistent homology of the simplicial complex obtained above is then computed with jHoles (Section 3.2, point 5 of the jHoles algorithm). The output is a collection of persistent barcodes, one for each DIM, and the set of the generators for the persistent topological features. An example of the output is given in Figure 5. This information is used for deriving the HDAs and for computing the persistent entropy.

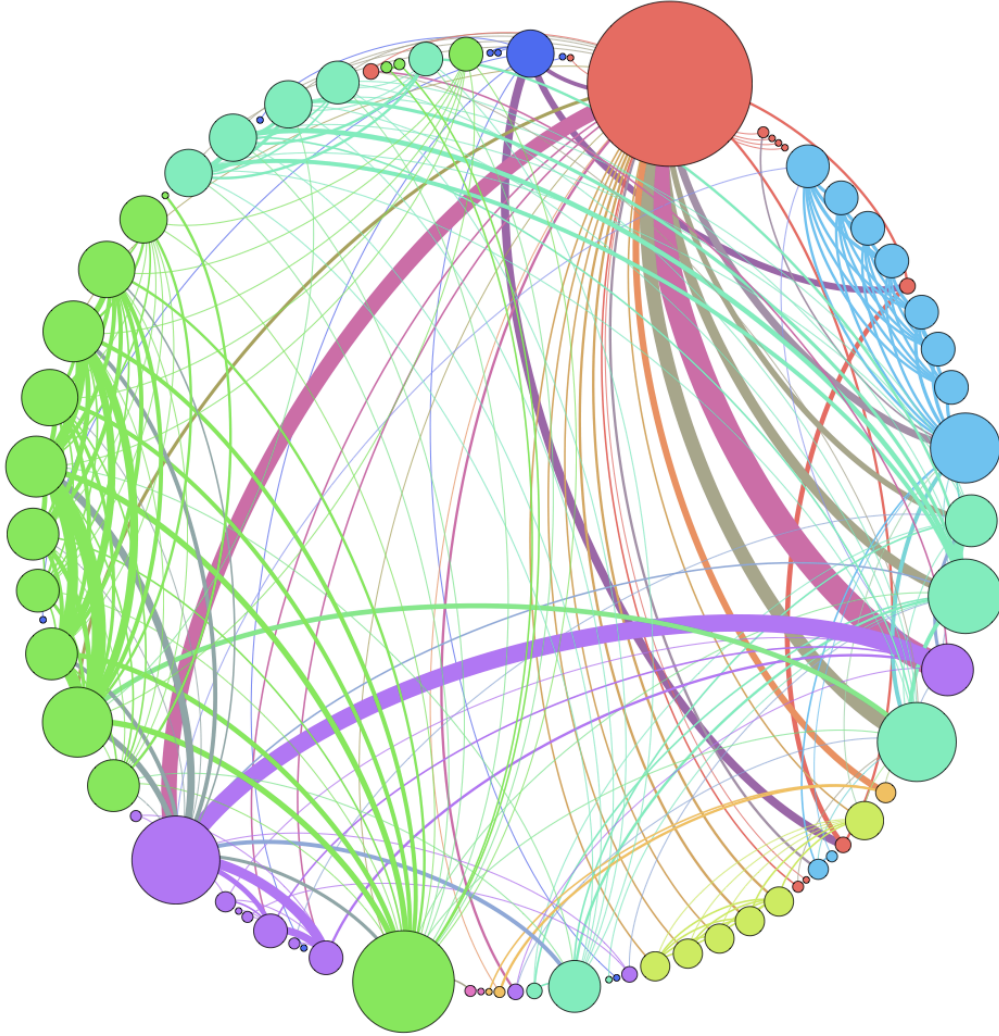


Figure 4: Example of immune network at the end of the simulation. The thickness of the arcs is proportional to their weight, the diameter of nodes is proportional to the number of incident edges.

4.3.1 HDA derivation

From the Jerne model (see Section 2) it is well known that antibodies exhibit a *true-concurrent behavior*, i.e., they can perform elicitation and suppression actions simultaneously on different targets. Using this domain knowledge we modeled the more relevant antibodies, given by the generators of the persistent topological features, with an HDA represented as a Chu space over the 3-length alphabet. A graphical representation of a Chu space is often given using an *Hasse diagram*. Hasse diagram helps to easily deduce

$$\begin{aligned}
& \beta_0: \\
& [0.0, \text{infinity}) : [16] \\
& \beta_1: \\
& [7.0, \text{infinity}) : [320, 3775] + [256, 3775] + [320, 3839] + [256, 3839] \\
& [6.0, \text{infinity}) : [256, 3839] + [256, 3711] + [384, 3711] + [384, 3839] \\
& [8.0, \text{infinity}) : [260, 3835] + [260, 3839] + [256, 3835] + [256, 3839]
\end{aligned}$$

Figure 5: Example of generators.

Ab	s1	s2	s3	s4	s5	s6	s7	s8	s9
(elicits, Ab , target)	0	0	0	1	1	1	2	2	2
(reduces, Ab , target)	0	1	2	0	1	2	0	1	2

Table 1: Chu space representation of an HDA modeling a generic antibody Ab performing two actions.

the set of admissible traces, dropping out automatically the *forbidden states*. In these diagrams the computation proceeds in the upwards direction and represents an execution path in the automaton [21]. As an example, Figure 6 shows the Hasse diagram representing the behaviour of a single antibody Ab executing two possible actions concurrently, whose corresponding Chu space representation is given in Table 1. Note that there is not a direct path between state $(1, 0)$ to $(0, 1)$ meaning that if action (elicits, Ab , target) is *executing* and action (reduces, Ab , target) is *unstarted*, then the first cannot go back to the unstarted situation while the second becomes executing (or vice-versa). Instead, from state $(1, 0)$ it is possible to go to state $(1, 1)$ meaning that both actions can be executing at the same time. Finally, if in the Hasse diagram the line between $(1, 0)$ and $(1, 1)$ was not present, that would mean that action (reduces, Ab , target) cannot start executing before action (elicits, Ab , target) has become *finished*.

As a more concrete example, we report the HDA modeling the behavior of two coupled antibodies Ab_1 and Ab_13 resulting from the simulations. Figure 7 shows the matrix of the Chu space representing such subsystem. Figure 8 shows the relative Hasse diagram.

4.3.2 Persistent Entropy Calculation

From the barcodes obtained in Section 4.3 we calculated, by using Definition 3.4, the value $H(t)$ of persistent entropy associated to every observation at time t . Note that in this case it is possible to build a chart of H over time because we know that the data are subsequent as they come from the same simulation. In general, for comparing two values of H one should define the so-called *mass function* [52].

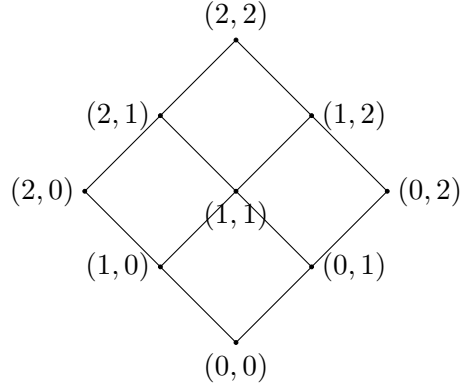


Figure 6: Hasse diagram for Chu space representing a generic antibody Ab_i

$$\{Ab_1 \mid Ab_{13}\}$$

$Ab_1 \mid Ab_{13}$	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}	s_{16}	s_{17}
$elicits Ab_{i,j}$	0	1	2	0	0	0	0	0	0	1	1	1	1	2	2	2	2
$reduces Ab_{i,j}$	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0
$elicits Ab_{j,i}$	0	0	0	0	0	1	2	0	0	1	0	2	0	1	0	2	0
$reduces Ab_{j,i}$	0	0	0	0	0	0	0	1	2	0	1	0	2	0	1	0	2

$Ab_1 \mid Ab_{13}$	s_{18}	s_{19}	s_{20}	s_{21}	s_{22}	s_{23}	s_{24}	s_{25}
$elicits Ab_{i,j}$	0	0	0	0	0	0	0	0
$reduces Ab_{i,j}$	1	1	1	1	2	2	2	2
$elicits Ab_{j,i}$	1	0	2	0	1	0	2	0
$reduces Ab_{j,i}$	0	1	0	2	0	1	0	2

Figure 7: Chu space representing the subsystem $\{Ab_1 \mid Ab_{13}\}$.

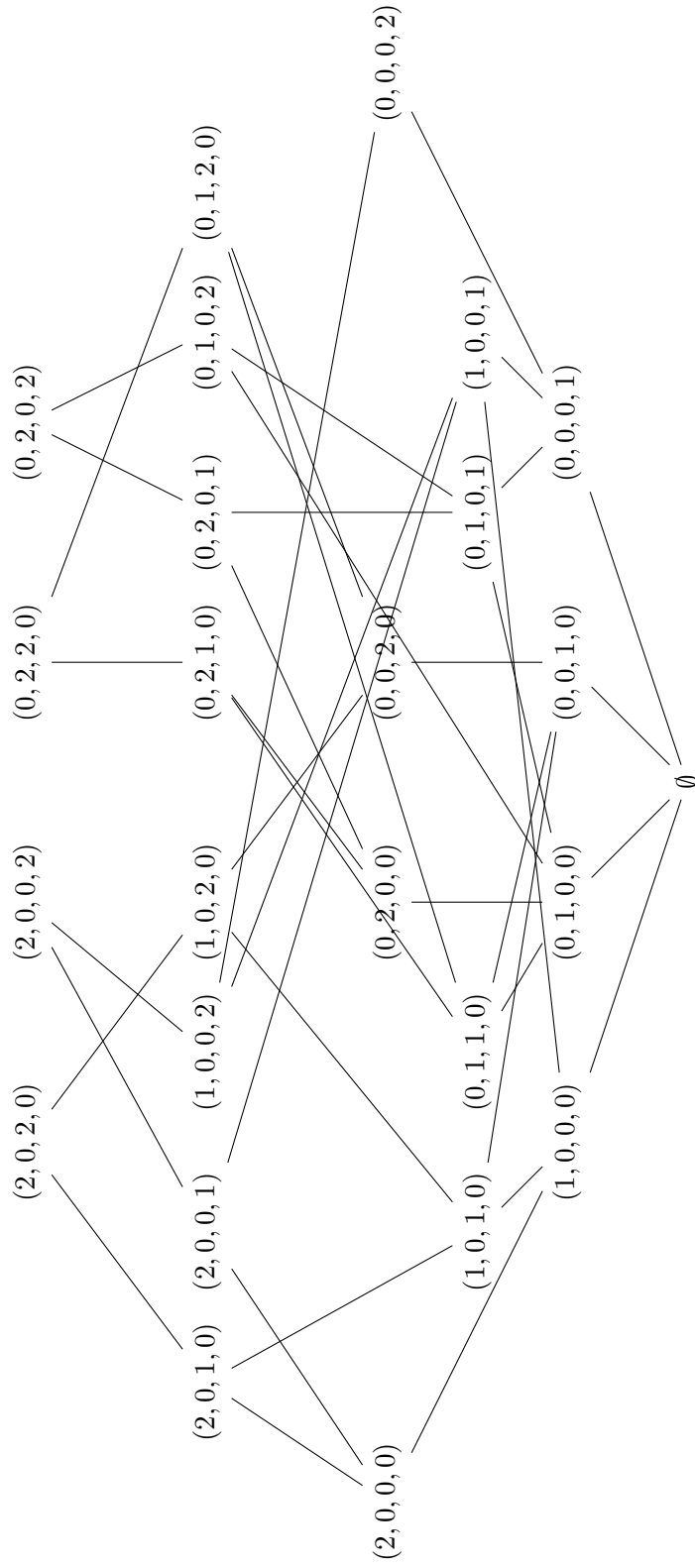


Figure 8: Hasse diagram for Chu space representing the subsystem $\{Ab_1 \mid Ab_{13}\}$.

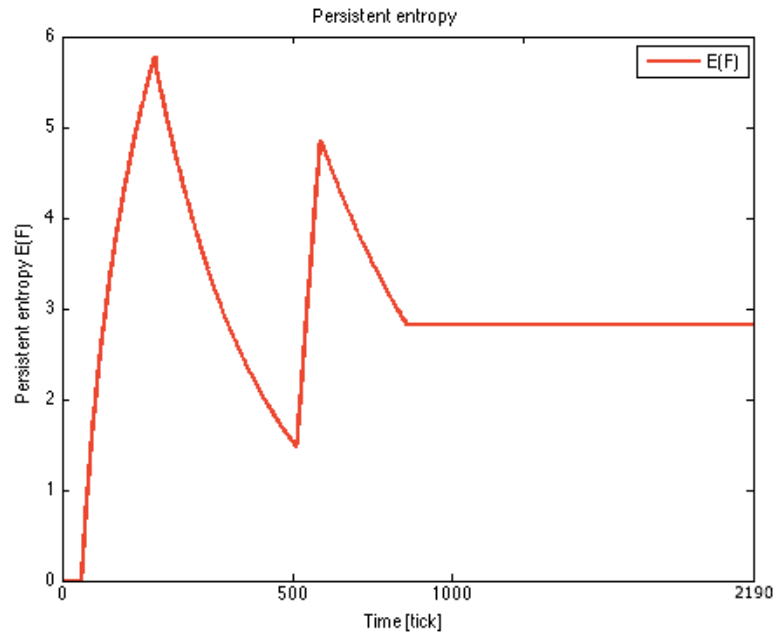


Figure 9: Persistent entropy of immune system. The difference between the peaks amplitude is motivated by the fact that before the second peaks the antibodies have been already stimulated and the *immune memory* has been reached, so the system is more reactive and is faster in the suppression of the antigen, thus the the entropy values for the steady state is $H=2.87$.

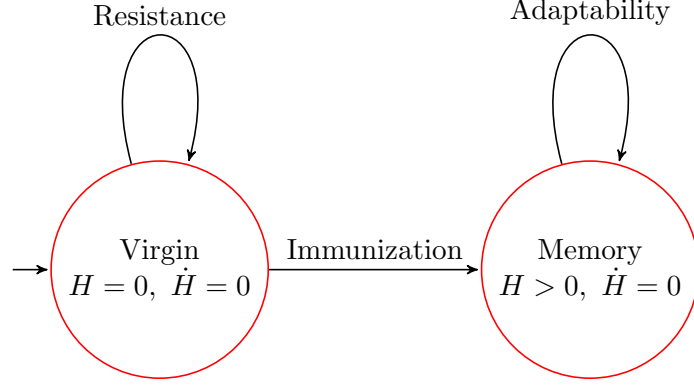


Figure 10: PEA representing the S level of the idiotypic network.

4.4 Analysis of the Plot of Persistent Entropy versus Time

Figure 9 show the charts of $H(t)$. Persistent entropy recognizes the dynamics of the immune system: a peak in the chart means *immune activation* while a plateau represents the *immune memory*. An upward curve between a peak and a plateau means *proliferation* while a backward curve means *immune response*. From the comparison of the charts it is possible to recognize that the system is stimulated twice (Figure 9).

Note that from the charts it is also possible to derive a discrete estimation of the first and second derivatives of $H(t)$ by using finite differences. These will be useful in the following section.

4.5 PEA Derivation

Analyzing the chart of persistent entropy we were able to recognize two steady states, namely *virgin* and *memory*. The *virgin* state of IN is characterized by an invariant condition $H = 0 \wedge \dot{H} = 0$. This is the initial state in the PEA that we derive. The second steady state corresponds to a plateau in the chart and it is characterize by an invariant condition $H > 0 \wedge \dot{H} = 0$.

Note that in the chart of PE (Figure 9) two peaks are present. This reflects the fact that the system was stimulated twice and, thus, entered an adaptation phase from state *memory* to itself. This then suggests that a self-transition should be added to the state *memory*.

Finally, even if we do not observe it in our simulation, we know from domain specific knowledge that when in *virgin* state, if the received stimulus is not grater than a certain threshold then the immune activation does not start at all and after a while the system goes back to the initial state again. This means that also in the initial state of the PEA a self-loop transition should be added.

5 Conclusions

In this work we defined a methodology, based on Topological Data Analysis, for deriving a sequence of two-level models, within the $S[B]$ paradigm, of a complex system from data. In particular, each data observation is initially represented with a weighted graph (network) and then its structure is used for building a richer topological space. Then, the topological space is characterized by computing persistent homology, from which local and global information about the complex system is extracted. The local information, about interacting computational agents, is represented by Higher Dimensional Automata. The global dynamics is studied by persistent entropy, which is newly introduced entropy measure linking together information theory and topology. We showed that the methodology can be effectively used by applying it to a case study about the idiotypic network of mammal immune system. As future work we plan to investigate the possibility of using Higher Dimensional Automata also for representing the global dynamics. In this way, we can improve the scalability of our methodology because an $S[B]$ derived for a certain system can be used as the model of an agent in the study of a larger complex system of which it is an interactive component.

Acknowledgments

We acknowledge the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme (FP7) for Research of the European Commission, under the FP7 FET-Proactive Call 8 - DyMCS, Grant Agreement TOPDRIM, number FP7-ICT-318121. The authors thank Mario Rasetti for being their mentor unique and invaluable during the years of TOPDRIM project and especially for his helpful suggestions but also criticisms.

References

- [1] Rajeev Alur and David L Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.
- [2] Jan A. Bergstra and Jan Willem Klop. Algebra of communicating processes with abstraction. *Theoretical computer science*, 37:77–121, 1985.
- [3] Marco Bernardo, Rocco De Nicola, and Michele Loreti. A uniform framework for modeling nondeterministic, probabilistic, stochastic, or mixed processes and their behavioral equivalences. *Information and Computation*, 225:29–82, 2013.
- [4] Marco Bernardo and Luca Tesei. Encoding Timed Models as Uniform Labeled Transition Systems. In *Proceedings of the 10th European Workshop on Computer Performance Engineering (EPEW 2013)*, volume 8168 of *Lecture Notes in Computer Science*, pages 104–118. Springer Berlin Heidelberg, 2013.

- [5] Massimo Bernaschi and Filippo Castiglione. Design and implementation of an immune system simulator. *Computers in Biology and Medicine*, 31(5):303–331, 2001.
- [6] Philip A Bernstein, Vassos Hadzilacos, and Nathan Goodman. *Concurrency control and recovery in database systems*, volume 370. Addison-wesley New York, 1987.
- [7] Jacopo Binchi, Emanuela Merelli, Matteo Rucco, Giovanni Petri, and Francesco Vaccarino. jHoles: A tool for understanding biological complex networks via clique weight rank persistent homology. *Electronic Notes in Theoretical Computer Science*, 306:5–18, 2014.
- [8] Nino Boccara and Nine Boccara. *Modeling complex systems*, volume 1. Springer, 2004.
- [9] Filippo Castiglione, Franco Celada. *Immune System Modelling and Simulation*. 286 Pages. April 7, 2015 by CRC Press ISBN 9781466597488
- [10] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
- [11] Gunnar Carlsson, Afra Zomorodian, Anne Collins, and Leonidas Guibas. Persistence barcodes for shapes. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 124–135. ACM, 2004.
- [12] Joseph Minhow Chan, Gunnar Carlsson, and Raul Rabadan. Topology of viral evolution. *Proceedings of the National Academy of Sciences*, 110(46):18566–18571, 2013.
- [13] Harish Chintakunta, Thanos Gentimis, Rocio Gonzalez-Diaz, Maria-Jose Jimenez, and Hamid Krim. An entropy-based persistence barcode. *Pattern Recognition*, 48(2):391–401, 2015.
- [14] Dipankar Dasgupta et al. *Artificial immune systems and their applications*, volume 1. Springer, 1999.
- [15] Vin de Silva and Robert Ghrist. Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology*, 7(339-358):24, 2007.
- [16] Xu-Tao Du, Chun-Xiao Xing, and Li-Zhu Zhou. Modeling and verifying concurrent programs with finite Chu spaces. *Journal of Computer Science and Technology*, 25(6):1168–1183, 2010.
- [17] Herbert Edelsbrunner and John Harer. Persistent homology a survey. *Contemporary mathematics*, 453:257–282, 2008.
- [18] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

- [19] Mohamed Fayad and Marshall P Cline. Aspects of software adaptability. *Communications of the ACM*, 39(10):58–59, 1996.
- [20] Robert Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.
- [21] Vineet Gupta. *CHU spaces: a model of concurrency*. PhD thesis, stanford university, 1994.
- [22] Lin Han, Francisco Escolano, Edwin R Hancock, and Richard C Wilson. Graph characterizations from von Neumann entropy. *Pattern Recognition Letters*, 33(15):1958–1967, 2012.
- [23] Thomas A Henzinger. *The theory of hybrid automata*. Springer, 2000.
- [24] Charles Antony Richard Hoare et al. *Communicating sequential processes*, volume 178. Prentice-hall Englewood Cliffs, 1985.
- [25] GW_ Hoffmann. A theory of regulation and self-nonsel discrimination in an immune network. *European journal of immunology*, 5(9):638–647, 1975.
- [26] John H Holland. Complex adaptive systems. *Daedalus*, pages 17–30, 1992.
- [27] John E. Hopcroft. *Introduction to automata theory, languages, and computation*. Pearson Education India, 1979.
- [28] Abasiofiok M. Ibekwe, Jincai Ma, David E. Crowley, Ching-Hong Yang, Alexis M. Johnson, Tanya C. Petrossian, and Pek Y. Lum. Topological data analysis of Escherichia coli O157: H7 and non-O157 survival in soils. *Frontiers in cellular and infection microbiology*, 4, 2014.
- [29] Lubomir Ivanov. Automatic Generation of Chu Space Model Expressions for Verification. In *Circuits and Systems, 2008. MWSCAS 2008. 51st Midwest Symposium on*, pages 613–616. IEEE, 2008.
- [30] Lubomir Ivanov. Modeling non-iterated system behavior with Chu spaces. *practice*, 12:24, 2008.
- [31] A Jankowski and A Skowron. *Practical Issues of Complex Systems Engineering: Wisdom Technology Approach*, 2014.
- [32] Nicholas R Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, 2001.
- [33] Niels K Jerne. Towards a network theory of the immune system. In *Annales d’immunologie*, volume 125, pages 373–389, 1974.
- [34] Jakob Jonsson. *Simplicial complexes of graphs*, volume 1928. Springer, 2008.

- [35] S Lakshmivarahan, Jung-Sing Jwo, and Sudarshan K. Dhall. Symmetry in interconnection networks based on Cayley graphs of permutation groups: A survey. *Parallel Computing*, 19(4):361–407, 1993.
- [36] E. Mancini; F. Castiglione; M. Bernaschi; A. de Luca A. and P.M.A Sloot: HIV Reservoirs and Immune Surveillance Evasion Cause the Failure of Structured Treatment Interruptions: A Computational Study, PLoS ONE, vol. 7, nr 4 pp. e36108. Public Library of Science, April 2012. (DOI: 10.1371/journal.pone.0036108)
- [37] Antoni Mazurkiewicz. Concurrent program schemes and their interpretations. *DAIMI Report Series*, 6(78), 1977.
- [38] Emanuela Merelli, Nicola Paoletti, and Luca Tesei. Adaptability checking in complex systems. *Science of Computer Programming*, 2015. DOI: 10.1016/j.scico.2015.03.004. Available online at <http://dx.doi.org/10.1016/j.scico.2015.03.004>.
- [39] Emanuela Merelli, Marco Pettini, and Mario Rasetti. Topology driven modeling: the IS metaphor. *Natural Computing*, pages 1–10, 2014. DOI: 10.1007/s11047-014-9436-7. Available online at <http://dx.doi.org/10.1007/s11047-014-9436-7>.
- [40] Emanuela Merelli and Mario Rasetti. Non locality, Topology, Formal Languages: New Global Tools to Handle Large Data Sets. *Procedia Computer Science*, 18:90–99, 2013.
- [41] Mikk, Erich and Lakhnech, Yassine and Siegel, Michael. Hierarchical automata as model for statecharts. *Advances in Computing Science*, pages 181–106, 1997.
- [42] Robin Milner. *Communication and concurrency*. Prentice-Hall, Inc., 1989.
- [43] James R Munkres. *Elements of algebraic topology*, volume 2. Addison-Wesley Reading, 1984.
- [44] Giorgio Parisi. A simple model for the immune network. *Proceedings of the National Academy of Sciences*, 87(1):429–433, 1990.
- [45] David Park. *Concurrency and automata on infinite sequences*. Springer, 1981.
- [46] G Petri, P Expert, F Turkheimer, R Carhart-Harris, D Nutt, PJ Hellyer, and Francesco Vaccarino. Homological scaffolds of brain functional networks. *Journal of The Royal Society Interface*, 11(101):20140873, 2014.
- [47] Giovanni Petri, Martina Scolamiero, Irene Donato, and Francesco Vaccarino. Topological strata of weighted complex networks. *PloS one*, 8(6):e66506, 2013.
- [48] Vaughan Pratt. Higher dimensional automata revisited. *Mathematical Structures in Computer Science*, 10(04):525–548, 2000.

- [49] Vaughn Pratt. Modeling concurrency with geometry. In *Proceedings of the 18th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 311–322. ACM, 1991.
- [50] Michael O Rabin. Probabilistic automata. *Information and control*, 6(3):230–245, 1963.
- [51] DL Stein and CM Newman. Nature Versus Nurture in Complex and Not-So-Complex Systems. In *ISCS 2013: Interdisciplinary Symposium on Complex Systems*, pages 57–63. Springer, 2014.
- [52] William J Stewart. *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling*. Princeton University Press, 2009.
- [53] Andrew Tausz, Mikael Vejdemo-Johansson, and Henry Adams. Javaplex: A research software package for persistent (co) homology. *Software available at <http://code.google.com/javaplex>*, 2011.
- [54] Glynn Winskel. *Event structure semantics for CCS and related languages*. Springer Berlin Heidelberg, 1982.